Volume 5, Issue 10, October-2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Survey on Smart Classifier Filesystem

Abhishek Pingale, Achilles Gaikwad, Soumitra Alate, Vishal Bembalkar
Department of Computer Engineering, SAE Kondhwa, Savitribai Phule Pune University (SPPU)
Maharashtra, India

Abstract—Imagine all your music, movies, documents are stored in the same folder, and you've to search every single time to find a specific file, among the cluster of this ambiguous data.Smart Classifier File System [SCFS] is a file system which classifies files automatically. More specifically, according to the file type and the metadata of the file. Using a file system which classifies files this way will allow the users to locate their files easily on the system. A classification based file system provides a means where files can be quickly and easily managed. Since users don't have to create the directories manually and save the data, this saves time. Traditional search tools are also another option for finding the files and locating them, but it is easier when you know where a specific file actually lies. Therefore it is always better to classify the files in a proper manner at the time of creation. A classification based content management system provides a means where data can be quickly and easily managed.

Keywords— Classification, Filesystem, Linux, Metadata, Smart, Inode, Musical tags, Document tags.

I. INTRODUCTION

Today people are relying on their computers and mobile phones for managing gigabytes of data that include personal emails, messages, documents, contacts, slides, audios etc. Managing and organizing such large amount of data is becoming difficult. It has become difficult for the user to traverse through number of files and folders and still not able to locate the file he needs because of numerous other files present in the same folder. The user has to search for the file he is looking for, but this uses one of the most important resources we need, Time. SCFS is a filesystem that classifies data by extending the inodes making the retrieval and the searching of the files as well as storing them easier for the end user. Its primary functionality is to automatically organize the files so that the retrieval becomes easy. SCFS provides function to perform automatic classification of files. Once a file is created, moved, or copied into a scfs partition, scfs automatically categorizes it into multiple pre-specified directories based on content similarity.

II. LITERATURE SURVEY

A. *Classification*

Classification is the category into which something is put. There are various ways of classifying a file, the file can be classified based on its file type, extension, user name, last access, modification and creation date, etc. Not all data is created equal, some are having higher priority for security while some are having higher preference for entertainment. Advantages of classification: The data which is classified can be easily found as the user knows where to find the data of a specific category. If the data is an audio file, the user can only look for the audio file, where the audio files are stored, rather than looking at the other irrelevant folders such as documents or videos.

B. *Hixosfs Music FS*

In Reference [1] the proposed filesystem is in linux kernel space. This filesystem has an architecture which is specially developed to support music. This is done by using widely used ext2 filesystem and extending the inode to store information like title, year, album, author/artist for musical data. It extends the inode struct inside the Virtual file system, considering as file properties Meta information such as album, author, and title related to the content of a musical file. This project required modified linux headers, new system calls (read and write tags), and the new hixosfs module to be implemented: *chtag, retag, chmusic, statmusic*. The linux header "linux/fs.h" has been changed so as to extend the definition of iattr struct with musical tag. Hixosfs extends the inode definition with a struct tag. This tag has four fields for a total of 100 byte of stored information. The new module which has been implemented includes certain functions like *hixosfs_new_inode*, *hixosfs_update_inode*and *hixosfs_read_inode*.

C. *Extended Attributes*

In[4] how a file can be managed using the *xattr* feature is mentioned.The file can be managed using metainformation. This is allowed by fs such as ext2, ext3, ext4, reiserfs. If the kernel is patched with the xattr attribute, it doesn't change the structure of inode physically. In *xattr*the attributes are stored as a couple (attribute-value) out of the inode as a string of dynamic length. The command that is used to deal with extended attributes in *xattr* is*attr*. This allows to set and get attribute values, remove attributes to list all of them, read/write these values to standard output.

D. Filesystem in userspace

The paper in[4]mentioned about a Filesystem in userspace FUSE, which is an operating system mechanism which allows non-privileged users to create their own filesystem. It is an opensource project with GPL and LGPL. FUSE acts as a bridge which provides access to the actual kernel interface. Virtual filesystems are written using FUSE. Virtual filesystems don't care about the data on the disk, they just arrange them to give a virtual view to the final user.

E. Knowledge File System

The paper in [3]shows a file system that helps users organize information using the familiar hierarchical file/directory contents. It is a virtual file system which occupies the space between the operating system and the file system. Features of this virtual filesystem are indexing, automatic classification of files and logging of usage. Once a file is created or copied in kfs partition then based on the content similarity it will be moved to the prespecified directories. Thus, Kfs helps user for faster retrieval of files by avoiding manual classification.

III. PROPOSED SYSTEM

A. Disadvantages of Existing File Systems

- Retrieval of files from the directories is difficult and takes large amount of time.
- There are very few filesystems that classify the files based on its extension during the file creation except the hixosfs and kfs which classify files, both of which are virtual filesystems.
- Virtual Filesystems require a FUSE which provides a bridge for users to communicate with kernel.

B. Proposed System Introduction

1) Modules

a) Disk Module -It consist of all physical devices which consist of magnetic media, motors and controls.

b) IO Supervisor Module - It is a module that supervises the interaction between file system request and input/output device drivers.[5]

c) Logical IO Methods - Thisconsist of device drivers which is a software program which communicates and control the operation of peripheral devices. [6]

d) Logical File Organisation Module - This module knows about the files and their logical blocks. This module is responsible for translating from logical to physical blocks. It maintains a list of free blocks and allocates the blocks to file as needed.[7]

e) File System Module – This module deals with retrieving and storing raw blocks of data.

f) User Process Module - This is module in which user will be communicating with the file system with the help of system calls.

2) System Architecture

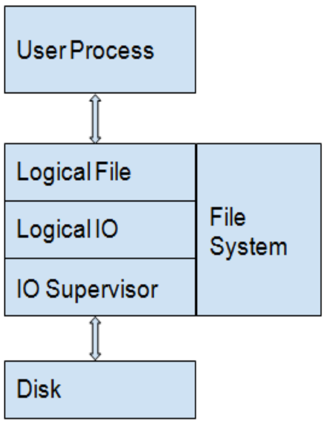

Fig. 1 System architecture of smart classifier file system

C. Mathematical Model

Let S be the solution perspective of the class
S={s, e, i, o, DD, NDD, success, failure}
Where, S is the start state
s=User mounts the filesystem
e = Unmounts the filesystem
Where, e is the final state
I = {i1,i2,i3,i4,i5,i6,i7,i8}
Where,
i1 = {create file}
i2 = {write to file}
i3 = {read file}

i4 = {open file}
i5 = {close file}
i6 = {lseek}
i7 = {change directory}
i8 = {list directory content}
i9 = {delete file}

Output = {s1,s2,s3,s4,s5,s6,s7,s8}
s1 = {created file}
s2 = {written to file}
s3 = {read file}
s4 = {opened file}
s5 = {closed file}
s6 = {changed the location of read/write pointer of file descriptor}
s7 = {changed directory}
s8 = {successfully list directory content}
s9 = {deleted file}
DD= {location of files with metadata and extensions such as media and documents}
NDD = {Files without metadata and extensions or files which aren't media or document files}
Failure= {file is not created, file is not opened, file is not deleted, directories or their content is not listed, file is not appended}

D. Algorithm

1. Start.
2. Create a file in temp directory.
3. Check the file type if it is media or document file and its metadata.
4. Check if the directory for a particular type of file or metadata of a file exists, if YES goto step 6 else goto step
5. Create a new directory according to its file type and metadata.
6. Copy file from temp directory to the particular directory.
7. Stop.

IV. FEATURES

The system we're proposing can classify all the documents, media files into appropriate folders. For example, if you've a song xyz.mp3 and a text file abc.txt which you place in SCFS, SCFS will smartly place the two files in directories reading their metadata. xyz.mp3 would be stored in
~/media/music/artist/album/xyz.mp3 while,
abc.txt will be saved in
~/documents/text/abc.txt

V. ADVANTAGES OF PROPSED SYSTEM

As aforementioned FUSE introduces a bridge from user to kernel, since SCFS is a filesystem in kernel space, there would not be use of FUSE eliminating the "bridge". SCFS will only require the metadata and the file extension for classification. It will classify the files based on their type, artist, albums, document/image creation dates, etc. Our system will be impeccable for the music files, video files, documents and images with metadata.

- The faster way to handle metadata is at kernel level.
- Can classify other file types such as documents, text files, video files, unlike hixosfs which only classifies multimedia data[2].
- Will have system call similar to statmusic present in hixosfs, which is better than *getXattr*as getXattr takes longer time to receive the attributes.
- Files stored can be easily browsed as it is categorized in a proper manner in a hierarchical way.

VI. CONCLUSION

In this paper we have analyzed different filesystems and methods which help in classification. We're concluding that since metadata can be handled faster in the kernel space, designing a filesystem that is in kernel space rather than user space is very effective for classification. The filesystem we are proposing will classify the user data, making it easier for specific users to make it efficient for searching and retrieval.We're going to eliminate the use of virtual filesystems which will reduce the latency to access the data. The data will be classified logically based on inode extensions. This will make the filesystem in kernel space better than a virtual filesystem.

ACKNOWLEDGMENT

We would like to take this opportunity to thank all the people who are a part of this Project in numerous ways, people who gave un-ending support right from the initial stage. In particular,We wish to Thank Mr. S.N. Shelke our internal project guide and Mr. K.S. Avhad our project coordinator who gave their coordination timely and precious guidance without which this project would not have been a success. We would like to thank our H.O.D Prof. B. B. Gite for his continuous encouragement and support and guidance at each and every stage of development of this project. And last but not the least I would like to thank all my friends who were Associated with me and helped me in preparing my project. The Project named Smart Classifier Filesystem which would not have been possible without the extensive support of people who were directly or indirectly involved in its successful execution.

REFERENCES

[1] Corriero, Hixosfs music: a filesystem in linux kernel space for musical files. MMEDIA 2009. home page.
[2] Corriero, Cozza, D. t. Z. (SIGMAP 2008). A configurable linux file system for multimedia data.
[3] Knowledge File System A principled approach to personal information management, Kuiyu Chang, School of Computer Engineering, Nanyang Technological University
[4] The hixosfs music approach vs. common musical file management solutions, Nicola Corriero, Vittoria Cozza, FabrizioFattibene.
[5] searchwindowserver.techtarget.com
[6] “The design of the unix operating system” by Maurice J. Bach.
[7] https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/11_FileSystemImplementation.html